Poster Title: ReBNet: Residual Binarized Neural Networks

Contributors: Mohammad Ghasemzadeh, Apple Inc.
Mohammad Samragh, UC San Diego
Farinaz Koushanfar, UC San Diego

Convolutional neural networks are widely used in a variety of machine learning applications, many of which are deployed on embedded devices. With the swarm of emerging intelligent applications, development of real-time and low-power hardware accelerators is especially critical for resource-limited settings (tinyML). The high computational complexity and memory footprint of these models are barriers to an efficient implementation. Leveraging qunatized values to represent the parameters of the model is a popular model compression technique to build an efficient accelerator. Binary neural networks, in which all weights and activations have two possible values, result in two particular benefits: (i) They reduce the memory footprint compared to models with fixed-point parameters; this is especially important since memory access plays an essential role in the execution of CNNs. (ii) They replace the power-hungry multiplications with lightweight XNOR operations. As such, Binary neural networks offer an intriguing opportunity for deploying large-scale deep learning models on resource-constrained devices (tinyML). However, binary networks suffer from a degraded accuracy compared to their fixed-point counterparts.

This work proposes ReBNet, an end-to-end framework for training reconfigurable binary neural networks on software and developing efficient accelerators for execution on FPGA. We show that the state-of-the-art methods for optimizing binary networks accuracy, significantly increase the implementation cost and complexity. To compensate for the degraded accuracy while adhering to the simplicity of binary networks, we devise the first reconfigurable scheme that can adjust the classification accuracy based on the application. Our proposition improves the classification accuracy by representing features with *multiple* levels of residual binarization. Unlike previous methods, our approach does not exacerbate the area cost of the hardware accelerator. Instead, it provides a tradeoff between throughput and accuracy while the area overhead of multi-level binarization is negligible. This work is accompanied by an API that facilitates training and design of accelerators.